


Just Enough Structure at the Edge of Chaos: Agile Information System Development in Practice

View metadata, citation and similar papers at core.ac.uk

brought to you by  CORE

provided by University of Queensland eSpace

Karimenz Kautz¹ and Sabine Zumpfe²

¹ Copenhagen Business School, Department of Informatics, Howitzvej 60, DK-2000 Frederiksberg, Denmark & University of New South Wales, School of Information Systems, Technology & Management, Sydney NSW 2052, Australia

Karl.Kautz@cbs.dk

² The University of Queensland, UQ Business School, Business Information Systems, Brisbane QLD 4072, Australia

S.Zumpfe@business.uq.edu.au

Abstract. Agile information systems development is not well understood and suffers from a lack of sustainable theories, which are based on empirical research of practice. We use a framework that focuses on the ‘edge of chaos’ as the area, where agile information systems development takes place to fill in this gap. Our study identifies for a concrete project under investigation, where the beneficial balance between stability and instability lies. It discusses the circumstances, which influence this balance and the relationships of the elements, which constitute it.

Keywords: Edge of chaos, complex adaptive systems theory.

1 Introduction

The field of information systems development (ISD) is still not well understood and suffers from a lack of sustainable theories which are firmly based on empirical research of ISD practice [1]. This is also true for agile information systems development or, to use the more established term, agile software development (ASD¹). The concept ASD serves as an umbrella for a number of pragmatic approaches which have emerged out of a critique of traditional, document driven development approaches [3]. ASD is guided by 4 values, which are contrasted with 4 other, competing values, namely (1) individuals and interactions over processes and tools (2) working software over comprehensive documentation (3) customer collaboration over contract negotiation and (4) responding to change over following a plan.

What this however means more concrete in practice and how it relates to a theoretical understanding of ASD as well as to ISD in general has only to a limited extent been systematically investigated and reported beyond text book descriptions and stories often provided by the authors of the methods themselves. The research

¹ The abbreviation ASD as used in this article should not be confounded with the same abbreviation, which Highsmith [2] uses for his agile development method called Adaptive Software Development.

presented in this paper contributes to further filling in this gap by providing an independent study based on scientifically collected and analysed empirical data.

For this purpose, understanding ISD as a complex adaptive system (CAS), it utilises a framework first introduced by Wang & Vidgen [3] that focuses on the investigation of the ‘edge of chaos’ in such a system as the area where ASD takes place.

The paper is structured as follows: The next section contains the theoretical framework, which is applied in our research. Section 3 includes both a brief description of our research approach and the case setting which built the background for our study. Section 4 analyses the collected data by applying the theoretical lens and presents our findings. The last section concludes and summarises the paper.

2 Theoretical Background and Framework

Some authors of ASD methods ([4], [5]) put forward that ASD has a theoretical grounding, namely in complex adaptive systems (CAS) theory. However, research ([6], [7], [8], [9]) has shown that this claim is largely a post-rationalization: the theory is, if at all, used in a very relaxed way to justify what is done in practice. Consequently the large amount of literature available on ASD is of anecdotal and descriptive character. While these are useful reports, they do not provide any deeper analysis or theoretical underpinning for a thorough understanding of ASD.

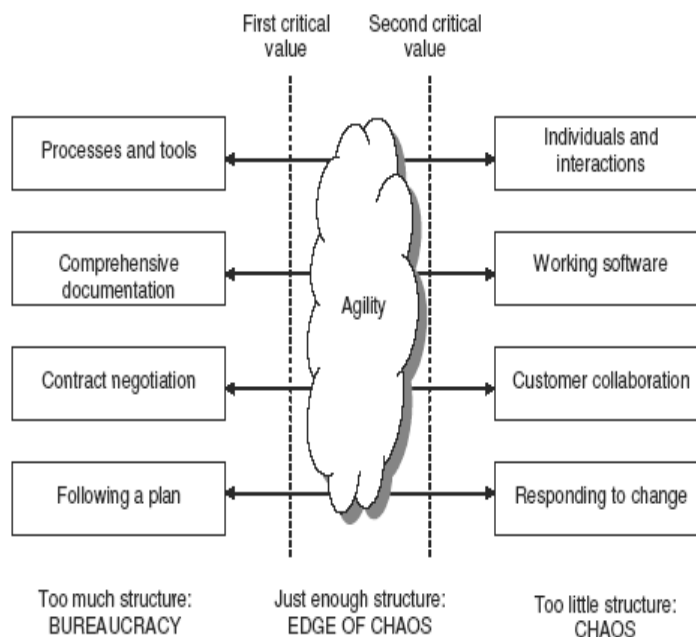


Fig. 1. The edge of chaos (Wang & Vidgen [2])

CAS theory, however, can be an insightful grounding for understanding complex systems such as ISD endeavours in general and ASD projects in particular. The key concept in CAS is the poise at 'the edge of chaos'. Wang & Vidgen [3] use this concept and provide a conceptual framework to study ASD as structured chaos. In their framework the edge of chaos is characterised by both being at the same time stable and unstable [10], it is the part of a system, which never quite locks into place, yet never quite dissolves into turbulence [11], it is the place, which provides organisations with both the stimulation and freedom to experiment and to adapt and with the sufficient frameworks and structures to avoid disorderly disintegration [12]. This gives them a competitive advantage: systems that are driven to the edge of chaos out-compete those which are not ([13], [14]). Thus, it is the place where the really interesting behaviour occurs [3]. Two critical boundaries demarcate the edge as the area of having 'just enough structure'. For the eight values, which direct ASD, this means: too much structure leads to bureaucracy with too rigid process and rules, too much documentation, too much emphasis on contracts and their negotiation, and too much focus on following the project plans; on the other hand, too little structure leads to chaos with too loose, if at all defined processes and rules, too much emphasis on working software, too much focus on collaboration, and too much response to change requests. Agility and agile processes lie some where in between, they are neither static nor chaotic. Figure 1 presents the framework and visualises these relations.

In the following we will use the framework to analyse a large ASD project. In doing so, we want to show in an independent study that the framework is useful. At the same time we want to contribute to the small, but growing existing body of scholarly research on ASD in practice. Before doing so, we introduce our research approach and the case setting.

3 The Research Approach and Case Setting

The research presented in this paper is qualitative. It is based on an empirical case study of an ASD project in a large German public sector organization, called WaterWorks, performed by a German software company, called AgDev, which has specialised on ASD. The empirical data for the case study was collected in semi-structured, open-ended interviews, which were conducted by a team of two researchers in a three days period. The research team performed 12 interviews with 11 individuals - the AgDev project manager was interviewed twice. This included nearly a third of the development team and a representative sample of key players and future users in the customer organisation. The interviews were tape-recorded and subsequently transcribed. For the qualitative data analysis a software tool (NVIVO7) was used. The interview data was supplemented with company and project documents such as method, requirements and release descriptions, as well as project plans.

The data collection, the coding of the data and the data analysis have been guided by the eight values underlying ASD and a theoretical framework developed and applied earlier by [15] to provide a better understanding of ISD in practice. This framework distinguishes between a structuralist, an individualist and an interactive process perspective, which together provide a holistic understanding of ISD projects. For the purpose of this paper we have used the structuralist perspective consisting of

the information system under development, the formalised method to be used (if any), the structural characteristics of the involved development team and its members, as well as the project's structural context, to identify the structural profile of the investigated project. This perspective helps us to structure the description of the case setting, which is summarized in table 1:

The project under investigation was concerned with the development of an operations management system (OMS) for the WaterWorks of a large German city. Founded 150 years ago the organization is now partially privatised with the city council holding 50.1% of the ownership. The system was developed with a web-based graphical user interface and a backend to interface the technical infrastructure as defined by an underlying ERP system.

The project was organized in 4 subprojects to provide IT support ranging from customer management to the maintenance of the sewer system. After several attempts of traditional ISD based on a standard ERP system, which had not led to the desired results, the organization opened a tendering process. It was won by a small software company, AgDev.

Table 1. The structural Profile of the OMS Project

Information System	Operations management system (OMS) with web –based GUI user interface and ERP back end
Formalized Method	xP: short releases and iterations of 3-6 months/3-6 weeks planning games, user stories, story cards, onsite users pair programming, collective ownership, stand-up meetings continuous integration, testing, re-factoring
Involved Development Team and Developers	2 overall project managers (1 AgDev, 1 WaterWorks) AgDev: up to 12 staff with multiple roles: project manager, analyst, customer contact, and developer highly motivated and educated, limited xP experience 4 subproject development leaders also as customer contacts, analysts, developers with xP experience WaterWorks: 4 customer subproject leaders also as user representatives at least 1 additional user representative for each subproject; not the whole time onsite
Structural Context	ASD method had been clearly communicated to customer AgDev: No experience with large ASD projects WaterWorks: 1. ASD project project team onsite in a WaterWorks building general requirements document as basis for contract failed ERP implementation

AgDev consisted of about 25 employees, 20 of them being developers, and based its development approach on the agile method xP [16]. The formalized method includes planning techniques for releases and iterations called planning games, user stories and story cards to specify user requirements, onsite customers to support customer-developer communication, daily meetings (stand-up meetings) of the whole project team to support team communication, pair programming, re-factoring,

collective ownership, continuous integration and testing to develop the software proper and tuning workshops to improve the development processes regularly. They have extended the method with some project management processes to cater for larger projects such an elaborate overall project plan, formal reporting mechanisms and a formal contract based on a requirements specification produced by the customer. In the tender process AgDev had convinced the management of WaterWorks that their approach was viable and would deliver the OMS as requested.

The project was organised in 2 phases. In a first 12 months exploration phase prototypes catching requirements and possible solutions were developed. This led to the development of a comprehensive requirements document by the customer organisation and their decision to contract AgDev also for the development of the OMS proper.

In this main development phase a team of about 12 development staff with multiple roles such as project manager, analyst, customer contact, and developer worked onsite in a building owned by WaterWorks. The project team also consisted of a varying number of users with at least one representing one of the subprojects. These users were by and large, however not the whole time onsite as well. A sophisticated management structure with one subproject manager also acting as contact person from AgDev and one subproject manager also acting as onsite-customer from WaterWorks for each individual subproject was, in addition to two overall project managers, established.

The developer team consists largely of highly educated and motivated, young staff and only the project managers have experience with ISD using an agile method, but none of them had ever participated in such a large project.

When this study was performed phase one had been successfully closed and after a break of over a year due to internal politics at WaterWorks phase two had been going on for 4 months. Responding to an inquiry call during our analysis the AgDev project manager stated that the project ended 10 months later on time and budget with all parts of OMS being operational. Despite some challenges (see section 4) from beginning to end the project was considered a success by all stakeholder groups involved.

The ASD method to be used had been clearly communicated to WaterWorks, which had been the main reason why AgDev won the tender. The project champion, an influential member of staff, who did not directly participate in the project team, but who was involved in most of the important decisions remembered: *“They presented a method, they explain it, and could convince us to get soon user feedback and a working solution.”* Thus it became WaterWorks’ first ASD project and the largest ASD project for AgDev at that point in time. With this structural profile in mind, we will now analyse the project in more detail.

4 Analysis and Discussion

The eight values are not just related to each others in pairs, but are highly interrelated. Processes and tools f. ex. beyond individuals and interactions, are also related to working software, customer collaboration, and responding to change; and individuals and interactions are also related to comprehensive documentation, contract

negotiations and following a plan. Due to the space limitations we will in the following stick to the pairing of values as put forward by the promoters of the agile manifesto to structure the presentation of our analysis and discussion and we will only use selected aspects of each value to demonstrate the usefulness of the ‘edge of chaos’ framework.

4.1 Individuals and Interaction over Processes and Tools

xP provides quite a number of processes and tools such as short releases and iterations, planning games, user stories, story cards, onsite users, pair programming, collective ownership, and stand-up meetings to structure ASD. In the OMS project pair programming is a prominent process to support the interaction of the individuals of the development teams by working in shifting pairs of developers in front of a screen while implementing the requirements written down as user stories on story cards as executable code. Two sub-processes or mechanisms here are important: 1) to regularly shift a partner and 2) to regularly shift possession of the keyboard within a team.

In the project the developers find it difficult to find the appropriate synchronization points at which to change a partner in the teams of 4 developers. No common practice exists. However they do not follow an overly bureaucratic rule such as shifting partner every morning regardless of the status of a story card. To avoid both too much red tape and too much chaos some developers prefer to stay with a partner until a card is closed. “... *changing a partner was always a problem, it still is as changing in the middle of a card seems foolish to me and I don’t really like doing it ...*” says one developer.

This of course can lead to limited interaction, spread of knowledge and dead ends. Thus, although some uncertainty regarding the mandate exists, a subproject leader might intervene if a pair has worked together for too long, say 3 days. In doing so, a balance, ‘just enough structure’ is created between shifting too often and not shifting at all.

The developers started out with a practice which did not really support the objectives of interaction, namely that one developer exclusively held the keyboard and programmed, while the other watched and sometimes commented. To avoid such situations a process was introduced where using a stopwatch after 20 minutes the keyboard had to switch. This was however abandoned as too bureaucratic and not fruitful in a creative work environment. The teams found their own rhythm. “*We don’t do that anymore. It didn’t function. Well, now it also functions without any explicit rule.*” was how one developer commented the emerged practice.

This has also been the case with stand-up meetings. They are performed by all teams together everyday before lunch with the purpose to keep everyone up to date with the current status of the project and to exchange useful information. These sessions originally were quite detailed and long, but they have been refined and were then acknowledged as very helpful. One AgDev subproject manager describes: “*In the beginning we did this all together, but we found out that it can become too much, as some are doing something that is not of interest for other teams. But it is good to know what others do. It does not have to be in detail. And that is what the teams do now, all teams, but we keep it short.*”

Other intensive interaction takes place in the beginning of each iteration, where all story cards are jointly discussed. Despite the fact that these mechanisms can not totally provide the intended collective ownership as the project leader regrets and explains with the size of the teams, they apparently provide enough structure for the project to be successful: they keep the project teams informed and decrease the need for documentation, a topic which will be discussed in the next subsection.

4.2 Working Software over Comprehensive Documentation

In the OMS project working software is the measure of progress. Each iteration produces operational software, but also minor advancements are demonstrated to the customers. The WaterWorks project manager stated: “... *I have never experienced a project that could generate output so fast.*” and continues “*The major benefit is that we do not work so abstract, but rather focus on the real thing.*” One of his subproject managers adds to this “... *this way we have seen that we are on the right way, as we can use 95% of what has been developed this way, and just the last 5% we have to do something with again ...*”. This is confirmed by one developer by saying “*Yes that functioned well, we made all 3 weeks a short presentation of the running software.*” and another one extended this: “...*we got very quick feedback when we showed what we had done.*” Thus, the short feedback cycles provide the necessary structure for the development of the working software. On the other hand, structure is also provided through documents.

Quite a number of different documents exist, but they are all comparably short and concise. From a customer perspective these are related as follows: “*Well, we have the overall realization concept as the basis for the contract and as a refinement hereof the requirements lists. These lists govern what should be the outcome of an iteration. For me this is the basis for my acceptance test: has been achieved what is on the list? And on the level below there are the story cards, these, so to speak, represent the detailed specifications and plans for the developers’ process.*” The developers share this perception and confirm that the documents, both in length and in number, are adequate. One of them says: “*Absolutely sufficient*” and is acceded by a colleague: “*I flipped through the realization document in the beginning and never touched it afterwards ... the requirements change anyway every 2. week.*”

The developers, however, also admit that they need and produce further internal documents, fragments of functional and technical specifications, in the form of an open Wikipedia and that there is a necessity to interact with the customers as part of their collaboration to clarify the contents of story cards. Together with these measures the utilized documents afford the balance between too little and too much structure.

4.3 Customer Collaboration over Contract Negotiation

Customer collaboration in the OMS project comes in different ways. It takes the form of onsite customers and users, as well as telephone contact and email correspondence, especially to clarify requirements as specified on story cards. The planning games, the presentations of working software and the acceptance tests are as well crucial elements, which structure the collaboration.

The planning games are partly based on the overall realization concept, a document which was produced by the customer as a basis for the contract. Another foundation of the planning game are the requirements lists. These are largely produced by AgDev, both their project leader and some of the subproject leaders, who also work as contact persons for their counterparts at WaterWorks and as developers. They develop these documents with input from the onsite customers. The story cards are solemnly produced by the developers in team work sessions, where they also estimate them. The developers and the customers then together prioritise these cards.

This can be considered a quite limited form of customer collaboration, however, as one subproject leader expressed it, there can be a number of reasons for this: *“Here we have users, who have to take their working gloves off before they go to the keyboard ... in contrast to projects I’ve been involved in before, where the customers were wearing ties, here the subproject leaders are partly folks, who have done something quite different before, they have a different education and that becomes apparent with regard to their abstraction capabilities and their abilities to write down some texts.”* However, this form of customer collaboration apparently provides some of the necessary structure to cope with the complexities of a comparatively large ASD project, which was performed by quite a number of inexperienced staff, while leaving room for less structured collaboration as well.

That is to say, when implementing the story cards, it became obvious that some additional collaboration was needed. One subproject manager estimates that user contact is necessary for nearly every story card. He puts forward that maybe 60% of a card’s contents is clear. When no user is onsite available the communication process is as follows: *“Certain users want to be contacted by phone to be reached straight away, while others prefer to get their requests via email, but answer timely.”*

The overall collaborative spirit of the project showing that the limited customer collaboration was not replaced by formal contracts and negotiations is also expressed by the project champion, who after having been involved in the original contract negotiations states: *“... we decided not to be tough on change requests and back-up formalities, but rather to work constructively with them to make progress. And my good feelings have been confirmed.”* The AgDev project leader confirms this and describes the context of requirement changes: *“The customer is quite relaxed. In such situations they look where they can cut expenses planned for other requirements or we discuss if we can make the implementation simpler to meet the budget planned.”*

The balance between stability and instability is brought about by different kinds of customer collaboration and by acknowledging, but not privileging the important role of contract negotiations, which also extends to the handling of change requests, which brings us to the last 2 values.

4.4 Responding to Change over Following a Plan

As described above, in the OMS project change, especially change of requirements is an accepted fact of life. Many change requests are detected through the scheduled acceptance test sessions for an iteration with a customer representative onsite and are then dealt with in the next iteration. The customer representatives also regularly

perform 'road shows' in the user departments to collect feedback and ideas and proposals for improvements.

But change requests are also brought forward by the users on a shorter time scale. There are weekly and bi-weekly feedback loops built into an iteration. The AgDev project manager explains: "*And then after a week the customer rep is back and wants to see what happened during the week and he gets the first feedback and this then continues*" They have the following consequence: "*... often we show the customer rep something once a week and then he's going 'well, I thought this would be different' ... thus there are always small changes ...*" as one developer puts it.

These frequent feedback loops also have the effect that minor misunderstandings are caught and dealt with as changes early before they can grow into something larger, as the same developer explains "*Until now it has not happened that everything was totally wrong; there are of course some refinements or a bug is found or something similar. There is always something.*" The feedback is taken seriously and immediately responded to with action: "*Through the feedback we got, we could react directly*" as it is described by one developer.

The different feedback mechanisms provide some structure to handle the changes, but plans and planning although not impeding more spontaneous actions are playing an important role as well. Even the weekly sessions are to some extent planned, as are of course the acceptance tests. As one WaterWorks subproject leader relating to the size and complexity of the project says "*Planning is essential in such kind of projects.*"

Therefore, the project also has an overall long term plan covering an 14 months period anticipating 3-6 releases depending on the subprojects. A more fine-grained plan is developed for the individual iterations, which make up a release detailed to single weeks. The planning game and the story cards then offer the devices to perform planning on the most detailed level for very short periods of time. The frequent planning sessions embedded in a 'larger' and coarser plan together with the different means to handle change provide just enough structure for the project to move forwards.

5 Conclusions

We have applied Wang & Vidgen's [3] framework to give a detailed description and analysis of how the edge of chaos provides just enough structure to perform a successful ASD project in practice. The framework takes both the preferred, and the less preferred values of the founders of the agile development movement into account. It shows that they are not opposites, but fundamental and interconnected elements, which, when balanced appropriately, present the ground for successful ISD endeavors. This balance will be different for different projects and more such studies are needed to identify common patterns, if there are any, beyond what has been found in this research.

Our study also shows how the 4 pairs of values are interrelated and how an analysis of these relationships provides a richer picture of practice as a prerequisite for understanding it and for building a sustainable theory of ASD and ISD.

6 References

1. Kautz, K.: The Enactment of Methodology – The Case of Developing a MultiMedia Information System. In: Proceedings of ICIS 2004, Washington D.C., USA, December 12 – 15 (2004)
2. Highsmith J.: Adaptive Software Development – A Collaborative Approach to Managing Complex Systems. Dorset House Publishing, New York (1999)
3. Wang, X., Vidgen, R.: Order and Chaos in Software Development: A Comparison of two Software Development Teams in a major Company. In: Proceedings of the 15th ECIS, St. Gallen, Switzerland, June 7-9 (2007)
4. Highsmith, J.: Agile Software Development Ecosystems. Addison-Wesley, Boston (2002)
5. Highsmith, J., Cockburn, A.: Agile Software Development: the Business of Innovation. In: Computer, 34(9), 120--122 (2001)
6. Kaleramo, J., Rissanen, J.: Agile Software Development in Theory and Practice, M.Sc. Thesis in Information Systems Science. University of Jyväskylä, Finland (2002)
7. Turk, D., R. France, R., Rumpe, B.: Limitations of Agile Software Processes. Third International Conference on eXtreme Programming and Agile Processes in Software Engineering. Alghero, Sardinia, Italy (2002)
8. Conboy, K., Fitzgerald, B.: Toward a Conceptual Framework of Agile Methods. Extreme Programming and Agile Methods - XP/ Agile Universe 2004, Proceedings, Berlin, Springer-Verlag, Berlin, Germany (2004)
9. Vidgen, R., Wang, X.: Organizing for Agility: a Complex Adaptive Systems Perspective on Agile Software Development Process. In: Proceedings of the 14th ECIS, Göteborg, Sweden, June 12-14 (2006)
10. Stacey, R. D.: Strategic Management and Organisational Dynamics: The Challenge of Complexity. 4th Edition, Prentice-Hall, London (2003)
11. Waldrop, M. M.: Complexity: The Emerging Science at the Edge of Chaos. Penguin Books, London, UK (1994)
12. McMillan, E.: Complexity, Organizations and Change. London, Routledge, Taylor & Francis Group (2004)
13. Anderson, P.: Complexity Theory and Organization Science. Organization Science: A Journal of the Institute of Management Sciences, 10(3), 216--232 (1999)
14. Kauffman, S.: At Home in the Universe: The Search for Laws of Self-Organization and Complexity. Oxford University Press, New York (1995)
15. Madsen, S., Kautz, K., Vidgen, R.: A framework for understanding how a unique and local IS development method emerges in practice. In: European Journal of Information Systems 15 (2), 225--238 (2006)
16. Beck, K., Andreas, C.: Extreme Programming Explained: Embrace Change. 2nd Edition, Addison Wesley Professional, Boston, Mass. (2004)